迭代器模式

> 指提供一种方法顺序访问一个聚合对象中的各个元素， 而又不需要暴露该对象的内部表示

- 迭代器的种类
  - 内部迭代器

    外界不需要关心迭代器内部实现, 跟迭代器的交互也仅仅是一次初始调用, 比如 forEach
  - 外部迭代器

    必须显示的迭代下一个元素, 实现起来稍微复杂一点, 但是灵活性比内部迭代器要高
- 实现自己的迭代器 (内部迭代器)

```
let each = function(array, callback){
    for(let i = 0; i < array.length; i++) {
        callback(array[i], i, array)
    }
}

each([1, 2, 3], function(item, index, array){})
```

- 实现一个外部迭代器

```
let iterator = function(array){
    let current = 0
    let next = function(){
        current+=1
    }
    let done = function(){
        return current >= array.length
    }
    let getCurrentItem = function(){
        return array[current]
    }
    return {
        next,
        done,
        getCurrentItem,
        length: array.length
    }
}

let it = iterator([1,2,3])
it.next()
it.getCurrentItem() // 2
it.done() // false
it.length // 3
it.next()
it.next()
it.done() // true
it.getCurrentItem() // undefined
```

- 作用

- 当需要迭代处理一个数组或者类数组对象时, 可以简化处理过程, 自动化处理相同的处理逻辑
- 简化迭代一个数据的操作, 使除了 for 语句外有了更多的选择